# Unix の Fortran 77 の互換性(数値解析と科学計算)

AUTHOR(S):

吉田, 肇; 長谷川, 武光; 真鍋, 篤

CITATION:

吉田, 肇 ...[et al]. Unix の Fortran 77 の互換性(数値解析と科学計算). 数理解析研究所講究録 1991, 746: 69-75

ISSUE DATE:

1991-03

URL:

http://hdl.handle.net/2433/102226

数理解析研究所講究録
第746巻 1991年 69-75



# UnixのFortran 77の互換性

吉田 肇　　長谷川 武光
福井大学工学部 情報工学科

真鍋 篤
高エネルギー物理学研究所 トリスタン計算機

## 1 はじめに

最近、強力なRISC CPUを使用したUnixワークステーションが急速に広がり始めている。それらはたとえば大型機M760とほぼ同じ計算能力を持っている。スーパースカラーCPUのスカラー速度はスーパーコンピュータのスカラー能力にひけをとらないほどである。CPU性能だけでなく、ダイナミックキャッシュなどの手法を用いることにより、ディスクスワッピングの効率を改善したワークステーションでは、大きな主記憶(約10MB以上)領域を要する行列計算においてはM760を凌ぐ。マルチプロセッサーのワークステーションでは、並列計算に使うにはまだまだであっても、複数のジョブのスループットの改善には目を見張るものがある。安価なワークステーションでも、入出力専用プロセッサを用いたり、入出力まわりのシステムソフトウェアを改善したりして、入出力の速度も大変に改良されている。そしてなによりも、Unixは大変優れたプログラム開発環境とネットワーク環境を提供している。Graphical User Interfaceなど、ヒューマンインターフェースの面でも優れた標準的な環境を与えてくれる。

Unixがもっとも得意とする分散環境の利用やXウィンドウなどのユーザインターフェイスはCでなければ有効には利用できない。Unixワークステーションの能力をもっともよく発揮するには、Fortranは適当でない。だが、高性能のワークステーションが100万円台で買えるようになった今、Unixだからワークステーションを使うというユーザ以上に、研究室で計算能力を調達したいからUnixを使うというユーザが増えてくる。その多くは、過去のソフトウェア資産の利用を考え、またCの数値計算ライブラリーがまだ限られている現状では(急速にこの状況は改善されつつあるものの)、やはりFortranを使用するであろう。

そこで問題となるのはFortranの互換性である。最新の計算機システムであろうと、20年以上にわたるFortranの財産を有効に使えなければならない。メインフレームの計算機でFortranを使っていたユーザが、最小の労力で、新しいUnixシステムに移行できることが望まれる。今まで、限られた数のメインフレームの計算機の間でも、Fortranの移植は必ず

しもスムースではない場合が多かった。両手で数え切れないほど多数の、強力なワークステーションがそれぞれ異なるFortran言語を提供し、研究室毎に好みのワークステーションを導入し、それらがネットワークで結ばれた場合、移植は今まで以上に大変なことになる。だが、Unixの標準にしたがっているならそのようなことは起きないのではないかという希望がある。

ところが、多数の最新のUnix計算機をベンチマークしている中で、この期待が裏切られることがあった。過去に書かれたプログラムが、何の問題もなくコンパイル·実行されるのに、実行結果は全く間違っているのである。結論から先にいうと、Backward compatibilityを保ちつつ、古くからのFortranプログラミングの慣習をどのように吸収するかという考え方の違いを、我々が注意深く読みとらなかったためであることがわかったが、このことが判明するまでにはすこし時間がかかった。新しくUnixに移行されるユーザの中には同様の経験をされる方がいるかも知れない。数値計算に直接関係することではないけれども、我々の経験を公開するのも無駄ではあるまいと考える。

## 2 何が起きたか

### 2.1 話しの始まり

我々の一人は、NEWS3860 (R3000, 4.3BSD) ワークステーションのf77 Fortranコンパイラで**aqlog.f**というプログラムをコンパイル·実行して、おかしな結果を得た。これは "An Algorithm Based on the FFT for a Generalized Chebyshev Interpolation" *Mathematics of Computation* Vol. 54, No. 189 pp.195–210 を実現した数値積分のプログラムである。既にUnixを含む他のシステム (MELCOM COSMO 700II, Sparcstation, Sun-3, Sigma-9100 , Titan II) では、どのシステムでも同様の正しい結果を得ていた。

### 2.2 事のひろがり

その後福井大学と高エネルギー物理学研究所で、ベンチマークする対象の機種を増やしたところ、MIPSが提供するf77 Fortranコンパイラを使っているシステムではすべて同様の現象が発生することが分かった。実際に確認したのは以下の5機種である。

**NEWS3860** Sony. R3000, 4.3BSD.

**RC3260** Kubota Computer (Stardent). R3000, System V + BSD.

**RC3240** 同上

**IRIS 4D/240S** Silicon Graphics. R3000×4, Syatem V + BSD.

**Decstation 3100** DEC. R3000, Ultrix.

MIPS f77 には最適化に関するコンパイルオプションが用意されている。そのひとつ-O2 (global optimization) では、レジスタの使用に関してコンパイラにバグが残っていたが、上述の結果は最適化のオプションがなんであれ発生した。

MIPS の f77 は cc (C コンパイラ) とは全く独立に作られているとのことである。Fortran よりもずっと多数多様の C プログラムをテストしたが、異常は見つからなかった。なお、AT&T の **f2c**(Fortran to C converter) を使って上記の **aqlog.f** を C のプログラムに変換し、C コンパイラでコンパイルして実行すると、正しい結果がえられた。

## 2.3 確かめのテスト

以下の計算機では正しい結果を得た。

**M780-2** FACOM

**MELCOM COSMO 700 II** Mitsubishi Electric. UTS/VS

**Sun-3/50** SUN. 68020+68881, OS3.5.

**Sigma 9100** Omron. 68030+68882, System V + BSD.

**Sparcstation** SUN. Sparc, OS4.0.

**Luna88K** Omron. 88K×4, Mach Release 2.5 (4.3BSD).

**MicroVAX** DEC. VMS.

**Titan II** Kubota (Stardent). vector computer, System V + BSD.

**C220** Convex. vector computer, 4.3BSD.

2 番めの計算機で Fortran IV を使った他は全て Fortran 77 に準拠しているコンパイラである。最初の二つ以外は全て Unix システムである。Unix に特徴的なことだが、採用している CPU も異なり、ベクトル計算機も含まれる。システムによっては、最適化オプションのあるなしで、倍精度数の 15、6 桁目あたりに若干の違いが見えるものの、上ののシステムも所定の要求精度にたいし収束する答を出した。

# 3 何が問題か

## 3.1 局所変数の扱い

**aqlog.f** プログラムには次のような部分があり、サブルーチン TRIGHP が始めて呼ばれた時にだけ、初期化の作業をするようにしている。MIPS の Fortran 77 コンパイラの既定の

コンパイルオプションでは、この制御が意図したようには動かない[1]。

```
      SUBROUTINE TRIGHP(M, ALPHA, COSPA, IER)
        ...
INTEGER ISW
DATA ISW /0/
IF(ISW .EQ. 1) RETURN
     ISW=1
       (初期化作業)
RETURN
END
```

Fortran 文法では、局所変数は手続きを出た時点でその値が保証されないとされている。そこで Fortran 77 では、局所変数をジョブの全過程中有効に出来るよう、**SAVE** 文が追加された。上の例では、2 回目に TRIGHP が呼ばれた時、局所変数 ISW の値が 1 である保証はない。

だが、上のような "古い" 手法を使っても問題が生じないよう、多くのコンパイラの implementations では拡張がなされている。たとえば、FACOM のフォートラン文法書、同使用説明書には、

"局所変数の値は手続き呼び出しを終っても**通常は保存される**ので、オーバーレイをする時以外は、**SAVE** 文を使う必要はない"

と書かれている。前の節で紹介した正しい答を出す多数派のシステムは同様の implementation であり、ユーザが黙っていれば自動的にこのような "拡張" が行なわれる。それにたいして、MIPS の f77 は、黙っているとこの "拡張" をしない。

## 3.2 Unix の標準はどうなっているのか

現在ワークステーションで採用されている Unix の主流である BSD と System V Unix の共通の出発点は AT&T の "Seventh Edition" である。"UNIX PROGRAMMER'S MANUAL" *Seventh Edition, Volume 2* 中の "A Portable Fortran 77 Compiler" *S. I. Feldman, P. J. Weinberger* の 3.5 節に次のような文章がある。(下線部は筆者。)

"A poorly known rule of Fortran 66 is that local variables in a procedure do not necessarily retain their values between invocations of that procedure. ... These rules permit overlay and stack implementations for the affected variables. ..."

ところが、実際の "Portable Fortran 77" の implementation f77 では、重要な仕様の拡張が行なわれている。特に、C 言語に似せて **static** と **automatic** 宣言文が追加された。こ

[1] このオプションのことを指摘して頂いたクボタコンピュータに感謝する。

れによりユーザは局所変数を **static** としてアドレスを割り付けるのか、**automatic** としてスタックに入れるのかを明示的に指定できるようになった。そのうえで、どちらを既定の設定とするかについては、2.5 節に、

"Local variables are static by default ; there is exactly one copy of the datum, and its value is retained between calls."

と明記されている。これが、"poorly known rule" を優先するのではなく、過去の慣習を尊重するための既定の設定である。この立場は昔からの Fortran ユーザにとってはありがたいものである。Unix であれば、BSD、System V、Ultrix などどれもこのマニュアルと同様の拡張と既定値の設定が行なわれていると考えるのは無理もないといえよう[2]。

## 3.3 MIPS f77 のコンパイルオプション

ところが、MIPS の Fortran 77 の立場は異なっていた。MIPS f77 コンパイラで、上の例のような "古い" コードを、意図する通りにコンパイルするには、**-static** というコンパイルオプションをつけなければならない。こうすれば、すべての局所変数は **static** 変数として扱われる。なお、Hewlett-Packard の 9000 シリーズの Unix Fortran も同様で、**-k** というオプションが用意されている[3]。なお、HP の場合少々複雑で、スカラー局所変数は黙っているとスタックに入れられるが、配列型局所変数は黙っていても **static** となる。

話しが少し脇にそれるが、RISC の場合は、スタックといっても必ずしも主記憶に置かれない場合がある。CPU 内部のレジスタに置くよう、コンパイラが再適化を行なうので、**automatic** 変数が主記憶上のスタックにあることを仮定してプログラムするのは危険である。

# 4 教訓

なるほど、MIPS f77 のマニュアルには **static** オプションの説明がちゃんとされている。われわれは、多数のシステムを次々とテストするために、自動的に作業が進むようにしたので、個々のシステム毎のマニュアルをきちんと読まなかったのは事実である。経験を積んだフォートランプログラマには、我々のあまりにも初歩的な経験は自明のことと笑ってすまされてしまうだろう。

なお、日本で出版されている Fortran 77 を扱っている教科書や参考書を数冊調べてみたら、Fortran 77 と歌いながら **SAVE** 文について一言も触れていないものや、簡単にしか紹介されていないものがあった。一般的に Fortran の implementations の互換性には神経を使

[2]SUN OS 3.5(Fortran Programmer's Guide の第 6 章、第 7 章) でも、全く同様のコメントが書かれている。

[3]福井大学工学部情報工学科都司達夫氏による。HP9000/300 シリーズの Fortran/9000 のマニュアルには、"変数の static allocation に依存して書かれた Fortran66 と 77 のプログラムを移植する時に使用する" よう書かれている。

うものだが、移植上の問題点を指摘したものはみあたらなかった。ネットワークにつながれた複数の計算機を同時に使用する場合が増えてきた現在、文法の教科書であってもCでやられているのと同様にもっと互換性について説明すべきではなかろうか。

## 4.1 互換性を重視した既定値とは?

以上が我々のささやかな経験であるが、問題が残るように思われる。よそで使っているプログラムを使用するために移植をするひとが、局所変数のことを良く知っていても、それが問題であることを見つけるるまでには少々時間がかかるのが通常であろう。MIPS (および HP) 以外のシステムのユーザ、特に大型機で Fortran 専用とも言える使い方をしているユーザは、"親切" なコンパイラの庇護の下にあって、上のような落し穴の存在を知らなくても生活できる。そのようなユーザが、最新の高性能の Unix システムへ移行をしようとする時、このような問題に出合うと、"Unix のフォートランは面倒だ" と、せっかくの Unix への移行の勇気がくじかれるのではなかろうか。RISC ワークステーションの性能価格比は素晴らしい。どんどん研究室レベルにひろがり、それとともに伝統的なフォートランプログラマが移行してくると予想される。そのような人たちが我々と同じ経験を繰り返すのは馬鹿げている。

MIPS が他のコンパイラと歩調を合わせて、既定のオプションを **static** としてくれるなら、当面の解決になる。実際、一社は、われわれの質問に対して **static** を既定のコンパイルオプションにする簡単な処方を提案した。

## 4.2 いくつかのメモ

多くの最新の Unix システムのフォートランをテストしていて、気がついたことがいくつかあるのでランダムにメモしておく。

1. 同じ会社の同じ Unix シリーズのシステムでも、機種により上述のオプションの既定の設定が異なっているものがある。分散ネットワーク環境では、このようなちょっとした違いが、たとえば、機種に合わせて **Makefile** を書き換えねばいけないなど、ユーザに手間をとらせる原因になる。

2. C に比べて Unix のライブラリの整備がまだ十分でない。とくに BSD と System V の違いが関わる部分が問題で、単に BSD 互換コンパイルオプションをつけたり、BSD ライブラリをリンクするだけでは駄目な場合がある。また、DARPA の標準的な関数をライブラリーでサポートしていないものもあった[4]。

3. 昔とは違い標準化が進み、どの CPU も IEEE 仕様の浮動小数点演算をしているといっても、数値計算の結果は同じではない。これは、関数や絶対値などの操作がコンパイ

[4] これは、コンパイラのドライバーの問題であって、メーカーからすぐに修正がなされた。

ラで異なっているせいであろう。

4. Vector 計算機ほどではなかろうが、最適化のオプションについては慎重に扱う方がよいと思われる。計算結果が最適化の有無でわずかながらも異なるものがおおい。数値計算の分野ではこれについてはさらに深い使用経験が要求されるだろう。また、コンパイラの optimizer にバグがあるものは黙っていないでバグレポートをユーザに報告したり、早急にバグ修正をするなどキチンとした対応をして欲しい。

5. 同じ問題を C と Fortran でプログラムして実行時間を比較した。計算を主とする場合は、C と Fortran の実行速度の違いはほとんどないといえる[5]。だが、ディスクの入出力が主な場合には C のほうがかなり早いワークステーションが多かった。

## 5 まとめ

Unix の最大の魅力の一つは、さまざまなハードウェアで共通の OS、プログラミング環境、ユーザーインターフェースが使用できることである。とはいえ、現実には OSF と UI に分かれてしまっているが、システムコールとはいわずとも、せめてコンパイラとそれの環境(ライブラリーなど)は早急に統一して欲しいものである。Unix にとって Fortran コンパイラーは決して新しいものではないが、多種多様なユーザがそれを使い始めたのは今までにはなかったことである。だからメーカーは“とりあえず Fortran もございます”というような態度はやめて今こそキチンとした方針を持って対処して欲しいものである。

[5]VAX Unix の Fortran が VMS に比べて非常に遅かったのは過去の話しのようである。